

The Seven Deadly Dangers

Of Outsourced Software Development

And How Companies Like Yours Can Avoid Them

by Steve Mezak, CEO, Accelerance, Inc.

Outsourcing your software development sounds like an ideal time and money saver, yet you may have heard of management horror stories, employee backlash and political nightmares. However, savvy software development departments are avoiding these issues and realizing the promise of outsourcing – decreased expenses and accelerated software development. This special executive report helps you successfully outsource your software development and avoid the *Seven Deadly Dangers of Outsourcing*.

Accelerance was founded in 2001 to be a prime contractor for global outsourced software development. Accelerance has many proven and expert engineering teams in over a dozen countries around the world. Accelerance helps its clients navigate past the seven deadly dangers with the strategies revealed in this report. With these strategies, you too can achieve reliable on-time, on-scope and on-budget results with risk-free outsourcing.

What is Outsourced Software Development?

Outsourced development is the hiring of a team of software engineers, usually along with a project manager, to develop your software product. Efficient communication and delivery via the Internet enables you to use outsourced teams anywhere in the world. Using an “offshore” development team is much less costly than hiring engineers in the US. This is especially true in an expensive neighborhood like Silicon Valley.

Contract outsourcing is the most common approach where a programming team is hired for months or even years to develop your software. Software companies that need a large team either create and staff a subsidiary immediately, or transfer an outsourced software development team into an offshore subsidiary after working with them for a year or more on a contract basis.



Danger # 1 – You Ignore Outsourcing

Ignoring outsourcing is not a realistic strategy. You and I both know outsourcing is not going away anytime soon. Use of outsourcing is growing rapidly. But here are some reasons why people choose to ignore outsourcing, at their own risk and peril:

- **You don't know how to select an outsourced team.** Its tough to select a team – there are many, many teams out there and choosing the right one is a challenge.
- **You think only a big project is worth the time & trouble.** You think you can just hire a few engineers for your project. But as we will see shortly, this can really add up and bust your budget.
- **You believe all your software must be developed “in-house” to get the best results.** Sometimes this is true. But why do you believe you HAVE to develop your code in-house? Is it fear of the unknown due to lack of experience with outsourcing?
- **You overlook the slow ramp-up to build your employee team.** Tell me, how long does it take you to hire a team of engineers? 2 or 3 months? That's a long time to wait until you start to write your software.

■ **You overlook the high burn rate to support your employee team.** Salaries are one thing. Then there is the cost of benefits and offices and computers, etc. Recruitment costs, benefits and overhead can easily add 30% to 50% or more to employee salaries.

Here are some numbers. Suppose you have a team of 20 engineers developing your software over a year, working as paid employees in your offices. The difference in a year's cost between in-house development (at a \$64/hour rate that includes benefits and overhead) & offshore outsourcing (at a relatively high \$32/hour) is about \$1 million. You are spending at least a million dollars more than if the engineers were offshore! So you see, deciding to outsource, or not, is truly a million dollar decision.

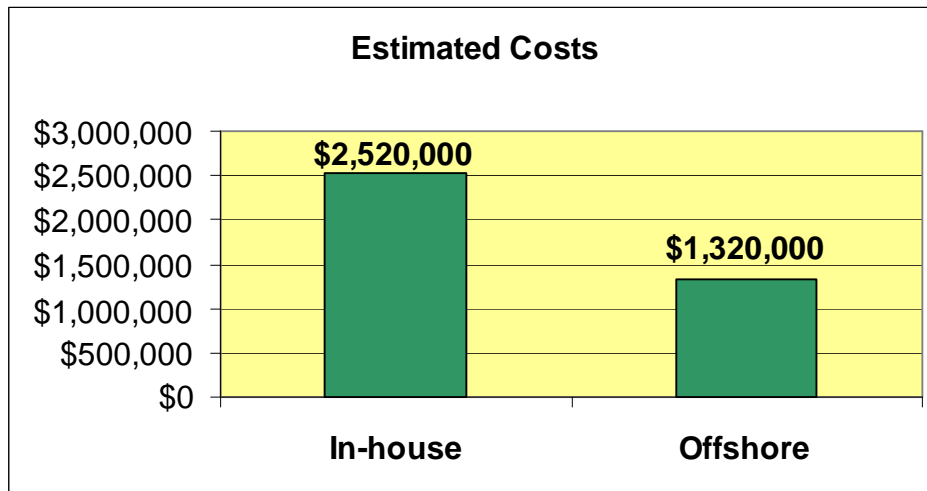


Figure 1- Estimated Cost of Twenty Programmers for One Year

The savings of offshore outsourcing are too large to ignore. Outsourcing is 25% to 50% of what it costs to develop software in-house. So, the question is NOT “Should You Outsource?” It becomes, “How Can You Make Outsourcing Work For You and Your Company?”



Risk-Free Strategy #1 – Make Your Optimum Outsourcing Decision

■ **Consider all your options.** Choose the best of 6 software development strategies for you.

1. **In-house** – Don't outsource. In some case doing in-house development is the best approach because of IP protection and other reasons, in spite of the higher cost.
2. **In-house/outsource blend** – of both in house and outsourced development teams working on the same code base.
3. **Onshore outsourcing** – to a software development company here in the US.
4. **Offshore outsourcing** – by far the most popular choice of American companies is to choose contract outsourcing to an offshore team in another country.
5. **Offshore subsidiary** – Jump in with both feet and commit to creating an offshore subsidiary right from the start.
6. **Offshore build-operate-transfer (BOT)** – gives you an option of creating a subsidiary and transferring the team into the subsidiary after a year or two of contract outsourcing.

■ **Use an analytical, objective tool** to help evaluate the business & technical issues of your situation.

Eighty percent of software companies are already outsourcing (According to [Software's Offshoring Leaders](#) published by Sand Hill Group). Should you? It's your decision and I am not going to try and convince you to outsource if you really don't want to. If you are not ready to outsource your software development then this report and Accelerance services are not for you.

But if your strategy is to drive down the cost of developing your software with outsourcing while increasing the predictability of your process then you came to the right place!

“OK,” you say, “I am convinced - I will start outsourcing right away!” But hold on. What about all those nightmare stories we hear about slipped schedules, wrong features, bugs and stolen IP? Why would anybody outsource if it is so risky!? What about the remaining 6 deadly dangers? Let's look at how to overcome each one.



Danger # 2 – You Hire the Wrong Outsourcing Vendor

Considering **ONLY your friend's roommate's brother in Bangalore or his cousin in Kiev** is unlikely to produce the “A” Team that best matches your software development needs. It is a common, and very human, mistake to look only amongst your immediate circle for resources. Similarly, being pressured into hiring the colleagues or friends of your investors puts the ease of creating a relationship ahead of the hard work of finding the best match for your project.

But it can take months to find and evaluate outsourced teams. That's why many companies avoid outsourcing to begin with. Here are some of the dangers encountered in selecting an outsourced team when you do it yourself.

■ **You don't diligently check multiple references before hiring.** When a recommendation is made by someone you know and respect, additional reference checks seem superfluous or even insulting.

■ **You scrutinize only the firm, not the individual developers assigned to your project.** Your outsourced development work will be done by a team of developers. The make up of the team and the strength of each individual will affect the quality and time-to-market of your software.

■ **You hire a team that is not dedicated to your project.** You're supplementing your workforce with an outsourced team to speed up the development process, not slow it down or

The VP of Engineering that carefully spent over 5 months making 9 site visits in 3 countries to select an offshore outsourcing vendor ...

... and he still got screwed

Recently, I met a VPE that carried out a careful search for an outsourcing vendor in India, China and South America. In the end, he narrowed his choices to 9 companies in India and China. He sought a BOT arrangement giving his company the option of transferring the team into a new offshore subsidiary sometime in the future. In the meantime, he needed a low price point to give him a specific cost savings over hiring engineers in the US.

He selected a US-owned outsourcing vendor with an operation in China. The vendor was in the process of acquiring a second team of programmers in China. The VPE found this second Chinese team to be excellent, and they quickly sketched out an architecture and design for the software that was needed. He then worked out the financial terms with the US-based vendor and signed the agreement.

Continued on the next page...

confuse it. An outsourced team that is over extended with other contracts, will not be able to deliver.

■ You hire a vendor that also sells software products. They will have a potential conflict of interest and may, knowingly or unknowingly, threaten the security of your Intellectual Property. Will your IP end up in their product? Will their IP end up in yours?

■ **You hire a vendor that is a “body shop.”** These are vendors that charge you for a room full of programmers but with adult supervision. You have to manage and direct their activities yourself. And that is difficult to do at a distance.

■ **You don't use a pilot project.** Experienced business people know, when embarking on any new business process, it is prudent to start with a small independent pilot to decrease the overall risk and the impact on your ultimate project.



Risk-Free Strategy #2 – Hire the Right Vendor

■ **Carefully check references** with other North American clients or English-speaking clients elsewhere. You should have at least two positive references and no negative ones.

■ **Hire only a team** that has a stable core of expert engineers. It is OK for them to hire a few new programmers to expand your team, but not more than 40% new hires as a guideline. Critical questions you should ask include: Are there junior members of the team who will get on-the-job training on your project; Is there frequent turnover – requiring someone (you?) to bring new members up to speed; Do the team members have the specific skills and experience to quickly deliver your software; Have the members worked together as a team before with outstanding results for other clients?

■ **Hire a team with several senior members** to guide and shepherd the development of your software.

■ **Look for a team with experience** developing software similar to what you need.

■ **Ensure that there is a core group on the team** that is focused exclusively on your software.

■ **Consider running a pilot project** for 3 to 6 weeks to confirm you are selecting a team that works well and meets your needs. A pilot allows a safe trial and a chance to move up the learning curve over a reasonable time period while minimizing risk. You can make and learn from small mistakes rather than large ones.

■ **Use a prime contractor, like Accelerance,** to save time and money, and the hassle of selecting your outsourced team selection on your own, and to manage the outsourced software development process. A prime contractor with connections to multiple outsourced teams can cut the time and cost of your selection process by as much as 90%.

But the acquisition of the second Chinese team by the US outsourcing vendor fell through. A junior team from the existing operation in China was assigned to the VPE's project.

Then the nightmare began.

English skills of the junior programmers were limited, making communication very difficult and inefficient. And their programming skills seemed worse. Their day-to-day activities had to be closely directed by the US-based VPE and his staff. The source code developed in China was reviewed daily.

Because of the 16 hour time difference with China, managers in the US spent many late nights emailing detailed instructions (even pseudocode) and answering questions by phone when it was daytime in China. This led to severe morale problems with the US staff, made worse by the fact that the US staff never thought outsourcing was a good idea to begin with. Missed deadlines and vociferous employee frustration eventually elevated the issue to the board level.

By the time I was called in by one of the venture investors, the VPE had begun turning things around. Executives from the American office of the outsourcing vendor “parachuted” in to help fix the problems. And after a long 6 month learning curve, the Chinese team finally became more efficient.

But the damage was done. Within two months, the VPE was gone. He resigned because the outsourcing engagement he so carefully arranged went bad, causing such an awful ruckus. He was forced to leave the company.

The [Accelerance Outsourcing Jumpstart service](#) enables your quick selection – weeks or days, rather than months – of an outsourcing vendor, based on your specific technical, business and strategic needs – saving you TIME & MONEY. You may already know an offshore vendor and feel like you don't need help looking for more. OK, but how many vendors do you know – two or three? The Outsourcing Jumpstart service draws on the many members of the Accelerance Global Partner Network but then quickly screens them down to the top three to five that are the absolute best fit for your situation – the best technology, best industry experience, best culture and methodology and the best time zone for you.



Danger # 3 – You Are Promiscuous with your Intellectual Property

The dangers of not emphasizing and protecting your unique intellectual property (IP) are multiplied when working with outsourced vendors. Here are some specific mistakes that put your IP in danger:

- **You put off developing and communicating IP protection policies and procedures** with your in-house team. It is one of those pesky procedural things that are often overlooked in the race to bring software to market. Setting up an outsourced project often highlights the lack of internal procedures for protecting your IP. Do your employees know what a “trade secret” is? Do they understand the value of your patent-pending software code and business processes? Do you have rules for who has access to key information and who shares what information with whom?
- **You are careless with employee non-disclosure agreements (NDAs).** When there are so many other priorities in the rush of software development, NDAs are often forgotten, or may even be considered an insult by core-team members. Having employees sign an NDA not only provides legal protection for the idea, it reminds them that the secret information of your company belongs to your company.
- **You have an outsourcing vendor that uses** computers with removable media that could create unauthorized copies of your software and IP.
- **You use an outsourcing vendor that transmits your source code** in clear text over the Internet.
- **You have no** written agreement with your outsourcing vendor. No non-disclosure agreements. No copyright assignments. No document that clearly states who does what and when payments should be made. This puts your IP at severe risk.

My employees would never...

Employees stealing software code? “Not MY employees,” you think. But it happens. In one case a former employee of Convergent Technologies started his own company. He took a booth at a Convergent tradeshow offering a cheap version of the operating system that he claimed he had “reverse engineered.” Fortunately, one of the original developers had buried a unique message deep in the source that said, “Hi, Irv.” When the suspect code was reviewed and the “Hi, Irv” message appeared, case closed, the source code had been ripped off.

Ensuring that you have clean, legal code is critical. You must understand and carefully document the pedigree of all source code used in your IP. If Open Source is included in your product, make sure you clean up any license issues before M&A due diligence. You could delay or devalue acquisition of your products or company because of suspicions over the ancestry of your product.

If you are creating a software product and ever want to sell your company, there will be no trail of ownership, no clear title to the software that is the basis of your company value.

- **You have an agreement but** you don't know if your vendor has the proper agreements with their employees. An employee that created your source could potentially leave and legally take your source code as his or her own creation!
- **You have critical IP in your source code**, but you don't divide or architect your software to isolate the most sensitive IP
- **You use open source software in your software without understanding license restrictions** both for use and geography. This may cause future legal problems with the ownership and sublicensing of your code and may impede your ability to reach global markets.
- **You assume your IP is safe** because your outsourcing vendor is in a particular country. Assuming a vendor is honest, without proper reference checks can lead to disaster.



Risk-Free Strategy #3 – Protect Your Intellectual Property

- **Hire an outsourced team** that is pro-active about protecting your IP. They must be committed to fierce protection of your IP as a normal part of their service offering. Review their established procedures for the protection of client IP.
- **Make sure you have physical protection.** You probably have some sort of physical security around your computers, servers and source code. Make sure your outsourced vendor has the same. The computers should be in a locked room, with card key access.
- **Make sure you have electronic protection.** Basic firewall, VPN and encryption technology is widely available to protect electronic assets like source code and electronic documents. Make sure both you and the outsourced vendor use them.
- **Architect your software for outsourcing.** When you outsource, keeping your sensitive IP isolated is key to ensuring your right to market your creative ideas worldwide. Dividing development of critical components between your in-house and outsourced team (or multiple outsourced teams) allows you to accelerate your development safely.
- **Make sure you have secure legal protection** – patents, trademarks and copyrights – at least in the US and the country of your outsourcer. For full protection you should register for IP protection in all the countries you intend to sell into, as the EU and other countries all have their own rules and requirements. No one entering a business relationship ever wants a dispute to go to court. Having a clearly stated agreement helps avoid disagreements later and helps you avoid the expense of legal resolution of disputes.
- **Use a prime contractor, like Accelerance**, based in the US to follow through every step of your outsourcing process to ensure your IP is protected to the maximum.
- **Make sure you have proper licenses** for any IP the outsourced vendor incorporates into your software. Software products are available now to automatically examine your source code and report on open source elements being used.
- **Sign an agreement with your outsourcing vendor** that covers the important points of IP ownership, copyright assignment, and non-disclosure of your source code and IP.
- **Get frequent updates of your source code** or place your source code repository in your control so your software cannot be held “hostage” if there is a dispute.

The [Accelerance Global Partner Network](#) members only develop custom software for clients. Vendors are evaluated using the critical IP protection requirements mentioned here. The [Accelerance Prime Contractor](#) service also provides you model agreements you can use to make sure your IP protection requirements are well understood by the vendor.



Danger # 4 – You Don't Know What Your Software Should Do

You don't know what your software should do, and lack of a specification makes it impossible to predict when your software will be completed. Programmers hate to write specifications. Yet having good requirements and specifications are key to successful software development, whether you outsource or not.

And don't forget about your hardware and performance requirements. Don't get caught by the old, "Didn't I tell you it has to run on a Macintosh?" problem.

■ **You code a detailed prototype instead of writing a specification.** This rarely results in robust software that solves your user's problems because you get caught up in the minutia of making the software work. You lose the bigger picture where you understand what your software should do to solve the problems of your users.

■ **You believe software cannot be specified ahead of time**, because writing software is a creative and innovative process. Then you hire an outsourced team anyway and hope for the best.

■ **You believe every detail of your software MUST be specified before outsourcing can be used.** It assumes you hand off a completely defined and encapsulated project to the outsourced team. While this removes risk of implementation delays it delays the start of coding. It also does nothing to accommodate changes.

■ **You believe outsourced engineers are drones** and cannot make any creative contributions to your software development. You ignore the valuable design and development advice available from experienced outsourced software development teams.

■ **You don't think through what should be developed in-house versus outsourced.** Your software should be architected for outsourcing so core development can be completed internally and the rest can be safely outsourced.

■ **You don't specify performance requirements or provide software use information.** Do you know how many users your software needs to support? How long should users have to wait to complete important operations? Do you provide details of low-level algorithms, but forget to explain how your customers will use your software? If you answer, "No", then there is not enough information to support outsourcing.

■ **You don't specify all the hardware and**

Protobuild - Implementation from Bottoms Up and then Top Down

Protobuild started implementing their construction management product from top and bottom. At the top, they had a graphics designer create user interface screens. They then added some content to illustrate how the product was to be used.

A local outsourced development team then implemented the bottom layer - the system objects, the relationships between them and the database tables used to store them. The combination was used to create a "live" prototype demo where screens were displayed using content drawn out of the database. But no complete scenarios or use cases were implemented and customer prospects could not see how the product would be used.

For example, you could display the budget for a construction project, but you could not make changes to the budget or create a new one. How would the system be used? It wasn't clear.

Then Accelerance was hired to bring in an offshore team. The underlying bottom layer objects and data model were mostly abandoned. Coding the objects within a product without a clear understanding of how they implement functionality is a mistake.

For example, the Protobuild product had projects with budgets, schedules and change orders. Should a change order be linked to an item in the budget? How about the schedule? We don't know until the actions users can perform are defined in the use cases of the product.

Work defining all remaining use cases was completed. Development proceeded to create a framework for session management and implementation of the basic use cases that were already well understood. Persistent entity objects and database tables were constructed as needed and re-factored through multiple releases as the requirements became better defined and understood.

Accelerance helped Protobuild complete their product by focusing on use cases – how customers will interact with the product. The offshore team was able to implement the product quickly given this top-down description.

systems that must be supported. Expecting the outsourcing firm to guess what you're thinking is sure to fail.



Risk-Free Strategy #4 – Be Clear about What Your Software Should Do

- **Move from idea to software** with a reasonable specification that captures the correct level of detail needed to develop the software effectively. Otherwise you get lost in the details of low-level coding issues instead of focusing on the needs of the customer. A clear software development plan is essential for successful outsourced software development. A complete plan specifying features, functions and performance allows any qualified software engineering team to create your software.
- **Clearly define your hardware platform requirements** to use outsourcing effectively.
- **Focus on defining the major use cases of your software.** Define how exceptions will be handled where understood. It is important for your software to respond gracefully with incorrect inputs and when errors occur.
- **Provide a high-level description** of the classes and objects you think are needed to implement your software. The additional details do not need to be specified initially, as they will become known as development proceeds.
- **Don't waste time by over-specifying your software** to levels of detail showing how it should be implemented. Stay at the higher level of defining how your software will be used. The right outsourcing team will have expertise in your technology and/or market area and will provide outstanding contributions to enhance your software.
- **Provide as much definition of the user interfaces as you can.** Be prepared to edit the wording and grammar in user interface elements created by an offshore team, if English is their second language.
- **Many firms use** outsourced teams with an agile approach where just enough requirements are provided to get coding started. This is described further in avoiding Danger #6.
- **Use a prime contractor,** like Accelerance, expert at designing software to help you prepare the proper level of specification required to make effective use of outsourcing.

The [Accelerance Rapid Ramp-Up](#) service helps you create a specification containing all the features, functions and benefits your product must have to be successful. Accelerance reviews any written materials you have created to describe your product and helps you evaluate the technical choices available for implementing your product. In addition to the specification, Accelerance provides cost estimates and a rough implementation time line for product development using outsourcing.

Time & Money Spent Onshore and Offshore, With and Without a Specification – A Comparison

Some people think it is a waste of time to write a specification. They would rather just use an in-house programming team and start coding to create the product as quickly as possible. The theory is a specification is not necessary because informal and direct communication should be sufficient when programmers work in close proximity with customers or a product manager. Here is an analysis that shows how this approach not only takes longer but is also more expensive whether you outsource or not.

We look at four scenarios, all using a single in-house product manager and 5 programmers that are either in-house or offshore. The first scenario produces a software release in 3 months of pure programming effort without a specification. The cost of the product manager and each programmer is \$10.5K per month, which includes salary, benefits and overhead.

The second scenario uses outsourcing with 5 programmers working offshore, also without a specification. This scenario is estimated to take an extra half month because of increased communication required. But each programmer costs only \$5.5K per month so overall cost is reduced 30% from development done in house.

	Spec Months	Coding Months	Product Manager	Engineers	Total Cost	Savings
In-house - no spec	0	3	\$ 31,500	\$ 157,500	\$ 189,500	---
Outsource - no spec	0	3.5	\$ 36,750	\$ 96,250	\$ 133,000	30%
In-house w/ spec	1	2	\$ 31,500	\$ 105,000	\$ 136,500	28%
Outsource w/ spec	1	2.5	\$ 36,750	\$ 68,750	\$ 105,500	44%

The third and fourth scenario has a US-based product manager spending a month gathering requirements and writing a specification before programming begins. Because of a clear specification, coding is estimated to take only 2 months with in-house programmers. An extra half month is allocated for additional communication when the offshore programmers are used in the fourth scenario

Clearly using a specification to direct your programming efforts will reduce the overall cost of creating your software. Offshore outsourcing is an additional cost saver even when extra time is added and a US-based product manager is used to guide the process to success.



Danger # 5 – You Use Meager Engineering Management

Unfortunately, you cannot completely rely on an offshore team to manage your software development. You can outsource the development but not the responsibility. After all, it is YOUR software. They will do their best to meet commitments to schedules and a high level of quality. But in the end, it will be your users trying to use your software. Here are how some companies go astray with outsourcing management:

- **You hire a programmer to manage software development.** This is a disservice to the programmer and to the team being managed. Management and programming are entirely different disciplines, requiring different training and expertise. Being good at one has no relationship to being good at another. The net result is an unhappy programmer and a frustrated outsourced development team.

■ **You leverage a scientist or CTO to manage software development.** This is worse than hiring a programmer for the role. The programmer may actually contribute to the release of the software. Scientists and CTOs often have a “timeless” quality about them, which can lead to delays or even to a lack of schedules to begin with.

■ **You use an incompetent manager to manage development** which delays the project and frustrates the team. It is critical that the project manager be able to make and keep commitments and ensure others keep theirs.

■ **You assign a person with limited technical knowledge** to manage software development. This leads to serious misunderstandings. The manager may be easily fooled by engineers, if the manager is unqualified to evaluate technical issues that arise.

■ **You don't give the appropriate authority to engineering management** to manage and approve work done by the outsourced vendor will lead to delays in completion of the project.

Lack of Authority Leads to Delays

A startup gave an engineering manager responsibility for directing the outsourced software development team, but gave him no authority over the team, resulting in delays.

The CEO negotiated complex terms with a US outsourcing vendor that included software licenses, small progress payments, royalties and stock. This deal was good for the CEO's company because it required little outlay of cash.

However, small cash progress payment offered little incentive for the outsourced team to meet schedule and manpower commitments. Many delays were encountered and the software quality suffered. The engineering manager did not know the exact details of the compensation plan and he also had little influence over the performance of the outsourcing vendor.



Risk-Free Strategy #5 – Use Effective Engineering Management

■ **If a scientist or subject matter expert** is required to provide innovations for your software, have someone work with them to translate the innovation into software specifications that can then be implemented.

■ **Your Engineering manager** must be goal-driven and focused on releasing the software on-time rather than the technology.

■ **Use a core technical team** with enough people to manage the various areas of technical detail of your software. They take responsibility for proper functionality and timely release of your software

■ **Provide effective supervision** to track scheduled milestones and releases.

■ **Have a member of the offshore team** work from your offices. This can be at the beginning, for short durations or the entire length of your project.

■ **Outsource your project management** to local resources that are experienced with outsourcing. They work closely with you on-site when needed.

■ **Use a prime contractor**, like Accelerance, to monitor the performance of your outsourced team and keep your outsourced software development on track.

The Accelerance [Outsourcing Prime Contractor](#) service provides the project management oversight necessary to ensure your software is developed on-time on-scope and on-budget using an outsourced team.



Danger # 6 – You Use a Mediocre Software Development Methodology

How do you go about the process of developing software? Do you create an excruciatingly detailed spec and then micromanage? Do you pile up the features for a single stupendous major release? And do you insist the offshore team cram all those features into the software by next Tuesday? If so, you have a mediocre software development methodology.

Do you assume “No News is Good News” if you have not heard from your offshore team? Do you strictly adhere to the strategy, “You code, I’ll design?” Do you not bother with QA until later? Do you NOT have a standard software release procedure or source code control system? If so, you have a mediocre software development methodology. Here are the dangerous details:

- **You outsource without a schedule of software release milestones**, leading to confusion and delays.
- **You dictate a schedule of software release milestones** to your outsourced team. Even if they cannot meet the schedule they will probably say “yes” anyway. If the outsourced team’s commitment seems too good to be true, it probably is. It is just a commitment to say what you want to hear.
- **You have just one major software release**, containing all the features. It creates a huge risk that keeps you “holding your breath” until the final release. It also keeps you from making the inevitable improvements and changes to requirements that come up to your software during development.
- **You dangerously use only infrequent communication with the outsourced team**, because you assume things are going OK. As with your in-house team, if you don’t ask they’re likely not to tell.
- **You make a strong (and artificial) distinction** between people that specify the software and those that develop it. This leads to a lack of teamwork and a breakdown in communication.
- **You rely on the heroic efforts of individuals to release your software on time**. This could create a disaster if a critical hero is unavailable or unwilling to deliver the extra effort required.
- **You don’t require unit testing of software modules** by your outsourced team and delays of your software release occur because changes to imperfect modules affect subsequent modules.
- **You don’t have a source code control system** and you have more than one programmer leading to confusion, errors and delays.
- **You don’t use coding standards** or require engineers to document their source code. This will create issues for both in-house and outsourced developers.



Risk-Free Strategy #6 – Implement an Effective Software Development Methodology

■ **Employ tools like UML and “use cases”** to define the structure and behavior of your software. These time-tested tools are a good way to describe what your software will do and helps define the milestones of functionality to be reached during development.

■ **Involve the outsourced team with the design of your software.**

You don't have time to completely design and specify the software. Sketch out the basic requirements and take advantage of the experience and skill of your outsourced team. They can help you get to a final design much more quickly and cheaply than you can on your own. And they learn the details of your software in the process.

■ **Let the outsourced team make an estimate** of the development effort. If the estimate is reasonable, have them commit to achieving it. Hold them accountable if the release date is not met. Accountability may take the form of a scolding and possibly withholding payment until the release is complete. Continued poor performance should lead to replacing the ineffective outsourced team.

■ **Have an acceptance test** that your software must pass before new code can be released as part of your software.

■ **Use an appropriate use of standards** and require documentation of your source code as it is written to facilitate quick ramp-up by new engineers and employees. This is essential in a dynamic world where code development is frequently collaborative and shared.

■ **To monitor progress, have frequent software builds** and release milestones with clear definition of the features and fixes

Agile & Collaborative Outsourcing

The “Waterfall” process of developing software has fallen into disrepute. Today, no one will gather requirements, write a specification, develop code and then test, as separate and distinct steps – each completed before the next one begins. It is a rare thing not to have your requirements change in the middle of development. Therefore, modern software development integrates these steps of design, development and testing together in each intermediate release. Together, a series of these releases incrementally build the functionality of your software.

For example, an Accelerance client created a specification for their software and began development with an Accelerance Affiliate in Russia. The 120 page spec described 6 major use cases and was copiously illustrated with screen shots from a prototype created in HTML. Even so, the spec was not complete in every detail. But coding began immediately to create an initial release rather than spend an inexorable amount of time creating a huge spec document.

Each morning the CTO would pour a cup of coffee and check email before heading into the office. A status report was received almost every morning with details of the development progress and problems discovered that day. Of course, “that day” in Russia was mostly our night in California.

Overlap of the Russian workday with the California morning, (you have even more overlap on the East coast of the US) was helpful in accelerating the completion of the software. Critical problems could be handled immediately while both Russia and California were on-line. Other less critical problems, or ones requiring more study, could be addressed anytime during the California day or evening.

Also development of this web application was done with frequent updates to a test version on-line. We could easily check the functionality over the Internet and report back problems, misunderstandings and issues by email during our work day. Problems were usually fixed by the next day.

The Internet enables this highly collaborative software development approach with outsourcing. Elements of Agile Development are used to start coding as quickly as some portion of the software is defined. It has been said that doing Agile Development is making yourself (and your outsourced software development team) useful until you understand what your software is really supposed to do.

targeted for each. It is better to have an intermediate release with new functionality that will not be shipped to customers, than to wait twice as long for complete functionality that is incorrectly implemented or too complex to test effectively.

■ **Have frequent communication.** This is essential to keep your development programs on track. Use a regularly scheduled on-line or phone conference to review status or require regularly submitted written status reports at least weekly in frequency.

■ **Consider using Agile Development methods,** to help get your software development started quickly and keep your development on-track with frequent builds and releases. Use a prime contractor like Accelerance that is expert in Agile Development methods to help guide your outsourced software development.

The [Accelerance Outsourcing Jumpstart service](#) delivers the vendors you need to take advantage of Agile techniques to create an open and visible development process with clearly defined milestones.



Danger # 7 – Your Quality Assurance is an Afterthought

QA is a critical part of the software development process. It is also a major concern when you outsource to programmers that are far away. Here are some dangerous mistakes companies make when they outsource that lead to poor quality:

■ **You use no bug tracking system** to keep track of defects, issues and enhancement requests. You are constantly tripping over bugs and repeat reporting of “known” bugs. A lack of process for fixing the bugs affects your overall software quality.

■ **You don’t specify the release criteria for your software** and create unclear expectations for the final release. Lack of these criteria may lead to misunderstandings of contract requirements and ultimately unsatisfactory releases of your software.

■ **You wait to start testing until just before you release your software** and rush an unacceptable product out the door. You have your customers or users find the bugs. These user experiences create negative word-of-mouth and press coverage that can doom a project, product and company.

■ **You think QA is what happens when users get your Beta release.** This does not create good will. User expectations for quality “out of the box” are increasing over time. Your reputation is damaged by bad Betas.

■ **You think unit testing by your outsourced team is sufficient.** No integration testing lets bugs and unusable software get shipped causing bad user experiences.

■ **You wait until automated testing can be implemented** to test your software. Then you find such a huge number of bugs causing extensive rework and delays in the release of your software.

■ **You don’t invest in enough computers to create a sufficient test and QA environment** which has a negative impact on overall quality and the performance of your software.

■ **You don’t test to verify load and performance requirements** causing your software not to meet your performance and capacity specifications required to honor user expectations. Customers may like your features and functions, but then they find your software does not work in “real world” situations.



Risk-Free Strategy #7 – Build in Quality Assurance

- **Specify your defect tracking system**, bug prioritization process and the release metrics you plan to use with your outsourcing team.
- **Confirm that your outsourced team** will do unit testing to find low-level defects during development.
- **Create an acceptance test** that your outsourced developers must run and pass before releasing software to you.
- **Use automated testing tools to create regression tests and test for performance and load-dependent bugs.** Use some amount of manual testing until your user interface becomes stable. The earlier a problem is caught, the less impact it will have on the remaining code being developed. A strong foundation is essential for high quality software.
- **Consider using outsourcing to a second team** to provide QA expertise and low cost manual testing resources if needed.
- **Use Alpha and Beta tests of your software** with customers and prospects to verify useful and beneficial functionality (not as a way of finding bugs). This keeps your customers in the loop and shows them you are responsive to their needs.
- **Use QA engineers to test your code on a variety of machines**, in a mix of scenarios and workloads, to ensure a good customer experience.
- **Use a prime contractor** like Accelerance that is expert in software development and QA, like Accelerance, to coordinate the over all software development process. A prime contractor can help you use a second outsourced team dedicated to QA.

The [Accelerance Outsourcing Jumpstart](#) service provides you the vendors experienced with the optimal strategies and tactics required for the manual and automated testing your product needs. Create a good impression with users by finding and fixing bugs before you release your software.

Outsourced Quality Assurance – To The Rescue!

One Accelerance client enthusiastically created a product without a QA plan. They started realizing their mistake as they encountered an increasing number of bugs themselves. So they decided to add a QA process a month before their Beta release.

Their product was a web application and could be tested with automated scripts. Even so, the amount of time it can take to find a qualified outsourced QA team can be excessive. Luckily, they knew about Accelerance. They were able to select a team that specializes in QA in one day and got them started immediately. The vendor found many bugs that were all fixed by the development team in time for the Beta release.

Implementing a QA plan is a great way to get started with outsourcing. You can take advantage of low-cost resources to this often neglected but critical part of the software development process. But do yourself a favor - plan on implementing a QA process as you begin your project. Don't wait until the very end!



Conclusion – Avoid the Seven *Deadly Dangers* & Take Advantage of Risk-Free Outsourcing



Use of outsourced software development has grown significantly over the last two decades and is accelerating. The Internet has enabled access to low-cost but expert and proven software development teams in many countries. Outsourcing is past the innovation and experimental stage, enabling you to make safe and effective use of outsourcing of your software development.

You don't need to repeat the mistakes made by software companies that have tried outsourcing in the recent past. You can benefit from their lessons learned and avoid the seven deadly dangers presented in this report. The BIG mistake is not using outsourcing due to fears of the associated risks, and not knowing about the methods and techniques that make it an effective tool to create your software.

Accelerance provides you an experienced, impartial source of information about outsourcing plus associated services that enable you to take advantage of risk-free outsourcing to develop your software.

© 2006 Steve Mezak, All rights reserved. You are free to use material from this article in whole or in part, as long as you include complete attribution, including live web site link. Please also notify me where the material will appear.

The attribution should read:

"By Steve Mezak, CEO of Accelerance, Inc. Please visit the Accelerance website at <http://www.Accelerance.com> for more information and resources on outsourced software development."



About the author: Steve Mezak is a successful Silicon Valley technical entrepreneur and international outsourcing expert. Get your copy of his free report now on "***Outsourced Development - A Secret Weapon for Quickly Creating Software Products***".

Get this valuable report now!

http://www.Accelerance.com/secret_weapon.htm